
pysimavr Documentation

Release 0.0.7

ponty

February 07, 2012

CONTENTS

1	Basic usage	2
2	Installation	3
2.1	General	3
2.2	Ubuntu	3
2.3	Uninstall	3
3	Usage	4
3.1	vcd export example	5
3.2	unit test example	6
4	File hierarchy	7
5	How to update simavr sources	8
6	API	9
6.1	low level interface	9
6.2	high level interface	15
7	Development	17
7.1	Tools	17
7.2	Install on ubuntu	17
7.3	Tasks	17
8	Indices and tables	19
	Python Module Index	20
	Index	21

pysimavr

Date February 07, 2012

PDF [pysimavr.pdf](#)

Contents:

pysimavr is a python wrapper for [simavr](#) which is [AVR](#) and [arduino](#) simulator

Links:

- home: <https://github.com/ponty/pysimavr>
- documentation: <http://ponty.github.com/pysimavr>

Features:

- python wrapper using [swig](#)
- [simavr](#) source code is included for easier installation
- object oriented interface on top of the generated interface
- maximum speed can be real-time
- serial communication
- check [simavr](#) documentation

Known problems:

- included [simavr](#) source code is not up to date
- Python 3 is not supported
- tested only on linux
- more tests needed
- PWM simulation is not real-time
- missing PWM modes
- a lot of messages on stdout
- LCD simulator is not fully implemented

Possible usage:

- unit test
- simulator

Similar projects:

- [simavr](#)
- [emulino](#)
- [Arduino Unit](#)
- [arduemu](#)

BASIC USAGE

```
>>> from pysimavr.avr import Avr
>>> avr=Avr(mcu='atmega48',f_cpu=8000000)
>>> firmware = Firmware('lcd.elf')
>>> avr.load_firmware(firmware)

>>> from pysimavr.sim import ArduinoSim
>>> print ArduinoSim(snippet='Serial.print("hello!");').get_serial()
hello!
```

INSTALLATION

check `simavr` doc: <http://gitorious.org/simavr/pages/GetStarted>

ignore these in `simavr` doc:

- OpenGL (freeglut)
- gcc-avr
- avr-libc
- make

2.1 General

- install `python`
- install `pip`
- install `swig` (for source build only)
- install header files and a static library for Python (for source build only)
- install a compiler (for source build only)
- install elf library
- install the program:

```
# as root
pip install pysimavr
```

2.2 Ubuntu

```
sudo apt-get install python-pip
sudo apt-get install swig
sudo apt-get install python-dev
sudo apt-get install gcc
sudo apt-get install libelf-dev
sudo pip install pysimavr
```

2.3 Uninstall

```
# as root
pip uninstall pysimavr
```

USAGE

pysimavr.examples.simple:

```
from pysimavr.avr import Avr
from entrypoint2 import entrypoint
```

```
@entrypoint
def run_sim():
    avr=Avr(mcu='atmega48', f_cpu=8000000)
    avr.step(1)
    print avr.pc
```

```
$ python -m pysimavr.examples.simple
Starting atmega48 - flashend 0fff ramend 02ff e2end 00ff
atmega48 init
2
```

pysimavr.examples.hello:

```
from pysimavr.sim import ArduinoSim
from entrypoint2 import entrypoint

@entrypoint
def run_sim():
    print ArduinoSim(snippet='Serial.print("hello!");').get_serial()
```

```
$ python -m pysimavr.examples.hello
Loaded 2202 .text
Loaded 26 .data
Starting atmega328 - flashend 7fff ramend 08ff e2end 03ff
atmega328 init
uart_udp_init bridge on port 4321
avr_timer_reconfigure-0 clock turned off
avr_timer_reconfigure-0 clock turned off
avr_timer_configure-0 TOP 7812.50Hz = 2048 cycles
avr_timer_configure-0 C 7936.51Hz = 2016 cycles
avr_timer_configure-0 TOP 976.56Hz = 16384 cycles
avr_timer_configure-0 C 992.06Hz = 16128 cycles
avr_timer_configure-1 TOP 30.52Hz = 524288 cycles
avr_timer_configure-1 C 7936.51Hz = 2016 cycles
avr_timer_configure-1 TOP 3.81Hz = 4194304 cycles
avr_timer_configure-1 C 992.06Hz = 16128 cycles
avr_timer_reconfigure-1 unsupported timer mode wgm=1 (0)
avr_timer_configure-2 TOP 976.56Hz = 16384 cycles
avr_timer_configure-2 C 992.06Hz = 16128 cycles
avr_timer_reconfigure-2 unsupported timer mode wgm=1 (0)
ADC Start AREF 0 AVCC 5000
UART-0 configured to 00cf = 4807 bps, 5 data 1 stop
Roughly 1666 usec per bytes
hello!
```

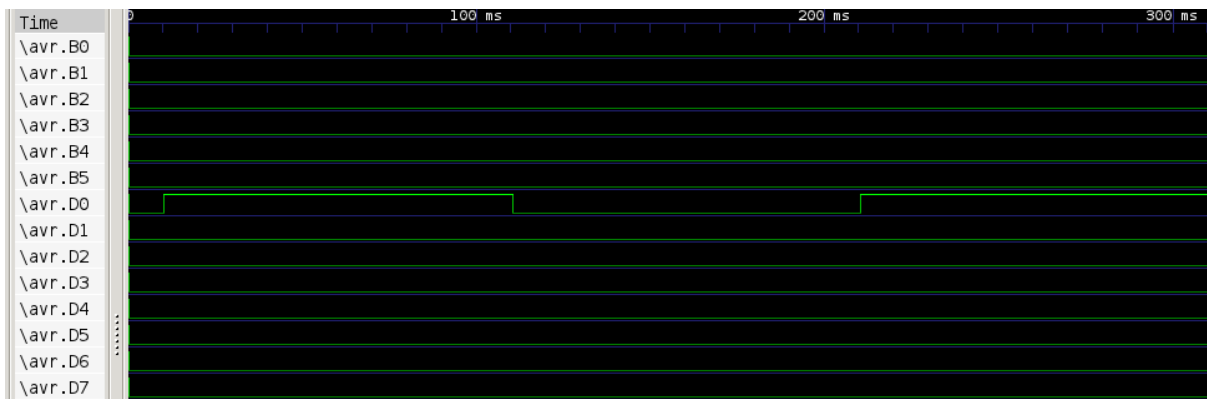
3.1 vcd export example

pysimavr.examples.vcd:

```
from entrypoint2 import entrypoint
from pysimavr.sim import ArduinoSim

@entrypoint
def run_sim(vcdfile='delay.vcd'):
    snippet='''
        Serial.println("start");
        pinMode(0, OUTPUT);
        digitalWrite(0, HIGH);
        delay(100);
        digitalWrite(0, LOW);
        delay(100);
        digitalWrite(0, HIGH);
        delay(100);
        digitalWrite(0, LOW);
        delay(100);
        Serial.println("end");
    '''
    sim=ArduinoSim(snippet=snippet, vcd=vcdfile, timespan=0.5)
    sim.run()
    print sim.serial

>>> from pysimavr.examples.vcd import run_sim
>>> run_sim(vcdfile='docs/vcd.vcd')
Loaded 2964 .text
Loaded 30 .data
Starting atmega328 - flashend 7fff ramend 08ff e2end 03ff
atmega328 init
uart_udp_init bridge on port 4321
avr_timer_reconfigure-0 clock turned off
avr_timer_reconfigure-0 clock turned off
avr_timer_configure-0 TOP 7812.50Hz = 2048 cycles
avr_timer_configure-0 C 12658.23Hz = 1264 cycles
avr_timer_configure-0 TOP 976.56Hz = 16384 cycles
avr_timer_configure-0 C 1582.28Hz = 10112 cycles
avr_timer_configure-1 TOP 30.52Hz = 524288 cycles
avr_timer_configure-1 C 12658.23Hz = 1264 cycles
avr_timer_configure-1 TOP 3.81Hz = 4194304 cycles
avr_timer_configure-1 C 1582.28Hz = 10112 cycles
avr_timer_reconfigure-1 unsupported timer mode wgm=1 (0)
avr_timer_configure-2 TOP 976.56Hz = 16384 cycles
avr_timer_configure-2 C 1582.28Hz = 10112 cycles
avr_timer_reconfigure-2 unsupported timer mode wgm=1 (0)
ADC Start AREF 0 AVCC 5000
UART-0 configured to 00cf = 4807 bps, 5 data 1 stop
Roughly 1666 usec per bytes
start
end
```

3.2 unit test example

pysimavr/examples/test_example.py

```
''' unit test example'''
```

```
from pysimavr.sim import ArduinoSim
```

```
def test_atmega88():
    mcu = 'atmega88'
    snippet = 'Serial.print("hi");'

    output = ArduinoSim(snippet=snippet, mcu=mcu, timespan=0.01).get_serial()
    assert output == 'hi'
```

```
$ nosetests --verbose pysimavr/examples/test_example.py
pysimavr.examples.test_example.test_atmega88 ... ok
```

```
-----
Ran 1 test in 3.117s
```

```
OK
Loaded 2120 .text
Loaded 22 .data
Starting atmega88 - flashend 1fff ramend 04ff e2end 01ff
atmega88 init
uart_udp_init bridge on port 4321
avr_timer_reconfigure-0 clock turned off
avr_timer_reconfigure-0 clock turned off
avr_timer_configure-0 TOP 7812.50Hz = 2048 cycles
avr_timer_configure-0 C 9345.79Hz = 1712 cycles
avr_timer_configure-0 TOP 976.56Hz = 16384 cycles
avr_timer_configure-0 C 1168.22Hz = 13696 cycles
avr_timer_configure-1 TOP 30.52Hz = 524288 cycles
avr_timer_configure-1 C 9345.79Hz = 1712 cycles
avr_timer_configure-1 TOP 3.81Hz = 4194304 cycles
avr_timer_configure-1 C 1168.22Hz = 13696 cycles
avr_timer_reconfigure-1 unsupported timer mode wgm=1 (0)
avr_timer_configure-2 TOP 976.56Hz = 16384 cycles
avr_timer_configure-2 C 1168.22Hz = 13696 cycles
avr_timer_reconfigure-2 unsupported timer mode wgm=1 (0)
ADC Start AREF 0 AVCC 5000
UART-0 configured to 00cf = 4807 bps, 5 data 1 stop
Roughly 1666 usec per bytes
```

FILE HIERARCHY

-docs	sphinx documentation
---_build	generated documentation
-pysimavr	main python package, high level classes
---examples	examples
---swig	all swig files (simavr and parts)
-----cores	copy from simavr
-----include	copy from simavr
-----avr	copy from avr-libc
-----parts	some electronic parts in c
-----sim	copy from simavr
-tests	unit tests

HOW TO UPDATE SIMAVR SOURCES

1. download simavr sources
2. download avr-libc sources (Ubuntu folder: /usr/lib/avr/include/avr/)
3. download pysimavr sources
4. copy over files:

```
$SIMAVR/include      -> $PYSIMAVR/pysimavr/swig/include
$SIMAVR/simavr/cores -> $PYSIMAVR/pysimavr/swig/cores
$SIMAVR/simavr/sim    -> $PYSIMAVR/pysimavr/swig/sim
$AVR_LIBC_INCLUDE/avr -> $PYSIMAVR/pysimavr/swig/include/avr
```

5. install pysimavr:

```
cd $PYSIMAVR
easy_install .
# or
pip install .
# or
paver install
# or
python setup.py install
```

API

There are 2 interfaces:

- `pysimavr.swig.*`: low level, generated by swig
- `pysimavr.*`: high level classes, they can redirect function calls to low level interface. Example: `Avr` class (high level) has all properties and methods of `avr_t` class (low level) automatically.

6.1 low level interface

```
class pysimavr.swig.ac_input.ac_input_t
```

```
    avr
    irq
    value
```

```
class pysimavr.swig.hd44780.hd44780_t
```

```
    avr
    cursor
    datapins
    flags
    h
    irq
    pinstate
    readpins
    vram
    w
```

```
class pysimavr.swig.inverter.inverter_t
```

```
    avr
    irq
    out
```

```
class pysimavr.swig.ledrow.ledrow_t
```

avr
irq
pinstate
pinstate_changed

class pysimavr.swig.sgm7.sgm7_t

avr
digit_count
digit_pin
digit_port
digit_segments
digit_segments_changed
irq
pinstate
segment_pin
segment_port

class pysimavr.swig.simavr.avr_io_t

avr
dealloc
ioctl
irq
irq_count
irq_ioctl_get
irq_names
kind
next
reset

class pysimavr.swig.simavr.avr_iopin_t

pin
port

class pysimavr.swig.simavr.avr_ioport_getirq_t

bit
irq

class pysimavr.swig.simavr.avr_ioport_state_t

ddr
name
pin

port

class pysimavr.swig.simavr.**avr_ioport_t**

io

name

pcint

r_ddr

r_pcint

r_pin

r_port

class pysimavr.swig.simavr.**avr_irq_pool_t**

count

irq

class pysimavr.swig.simavr.**avr_irq_t**

flags

hook

irq

name

pool

value

class pysimavr.swig.simavr.**avr_kind_t**

make

names

class pysimavr.swig.simavr.**avr_symbol_t**

addr

symbol

class pysimavr.swig.simavr.**avr_t**

aref

avcc

codeend

cycle

cycle_timer

cycle_timer_map

data

e2end

eind

```
flash
flashend
frequency
fuse
gdb
gdb_port
i_shadow
init
io
io_port
io_shared_io
io_shared_io_count
irq_pool
log
mmcu
next_cycle_timer
pc
pending
pending_wait
ramend
rampz
reset
run
signature
sleep
special_deinit
special_init
sreg
state
trace
trace_data
vcc
vcd
vector
vector_size
class pysimavr.swig.simavr.avr_t_cycle_timer

    param
    timer
```

```
    when
class pysimavr.swig.simavr.avr_t_io

    irq
    r
    w
class pysimavr.swig.simavr.avr_t_io_r

    c
    param
class pysimavr.swig.simavr.avr_t_io_shared_io

    io
    used
class pysimavr.swig.simavr.avr_t_io_shared_io_io

    c
    param
class pysimavr.swig.simavr.avr_t_io_w

    c
    param
class pysimavr.swig.simavr.avr_trace_data_t

    codeline
    old
    old_pci
    touched
class pysimavr.swig.simavr.avr_trace_data_t_old

    pc
    sp
class pysimavr.swig.simavr.avr_vcd_log_t

    signal
    value
    when
class pysimavr.swig.simavr.avr_vcd_signal_t

    alias
    irq
    name
```


size

class pysimavr.swig.simavr.**avr_vcd_t**

avr

filename

log

logindex

output

period

signal

signal_count

start

class pysimavr.swig.simavr.**elf_firmware_t**

aref

avcc

bsssize

codeline

codesize

command_register_addr

console_register_addr

datasize

eeprom

eesize

flash

flashbase

flashsize

frequency

mmcu

trace

tracecount

tracename

traceperiod

vcc

class pysimavr.swig.simavr.**elf_firmware_t_trace**

addr

mask

name

6.2 high level interface

```
class pysimavr.ac.Ac (avr)
```

```
    getirq (pin)
```

```
class pysimavr.avr.Avr (firmware=None, mcu=None, f_cpu=None, avcc=5, vcc=5)
```

```
    arduino_targets = ['atmega48', 'atmega88', 'atmega168', 'atmega328p']
```

```
    avcc
```

```
    fpeek (addr)
```

```
    getirq (pin)
```

```
    goto_cycle (n)
```

```
    goto_time (tsec)
```

```
    load_firmware (firmware)
```

```
    move_time_marker (tsec_diff)
```

```
    pause ()
```

```
    peek (addr)
```

```
    reset ()
```

```
    run ()
```

```
    states = ['Limbo', 'Stopped', 'Running', 'Sleeping', 'StepStepDone']
```

```
    step (n=1, sync=True)
```

```
    terminate ()
```

```
    time_passed ()
```

```
    vcc
```

```
exception pysimavr.avr.UnkwownAvrError
```

```
pysimavr.connect.connect_irqs (irq_out, irq_in, bidirectional=False)
```

```
pysimavr.connect.connect_pins_by_rule (rule, device_map, vcd=None)
```

```
    rule example:
```

```
    B0 -> D4 -> vcd
```

```
    B1 <== D5 B2 ==> D6 #B3 <=> D7
```

```
class pysimavr.firmware.Firmware (filename=None)
```

```
    mcu
```

```
    read (filename)
```

```
class pysimavr.inverter.Inverter (avr)
```

```
    getirq (pin)
```

```
    out (i)
```

```
class pysimavr.lcd.Lcd (avr, size=(20, 2))
```

```
    get_char (x, y)
```

```
getirq (pin)  
pinstate (pin)  
reset ()
```

```
class pysimavr.ledrow.LedRow (avr, size=8)
```

```
getirq (pin)  
pinstate (i)  
reset_dirty (i)  
    read and reset
```

```
class pysimavr.sgm7.Sgm7 (avr, size=4)
```

```
digit_segments (digit_index)  
getirq (pin)  
pinindex (pin_name)  
pinstate (pin)  
reset_dirty (digit_index)  
    read and reset
```

```
class pysimavr.vcdfile.VcdFile (avr, filename='gtkwave_output.vcd', period=10)
```

```
add_signal (irq, name=None, bits=1)  
start ()  
stop ()  
terminate ()
```

DEVELOPMENT

7.1 Tools

1. `setuptools`
2. `Paver`
3. `nose`
4. `ghp-import`
5. `pyflakes`
6. `pychecker`
7. `paved fork`
8. `Sphinx`
9. `sphinxcontrib-programsscreenshot`
10. `sphinxcontrib-paverutils`
11. `autorun` from `sphinx-contrib` (there is no simple method, you have to download/unpack/setup)

7.2 Install on ubuntu

```
sudo apt-get install python-setuptools
sudo apt-get install python-paver
sudo apt-get install python-nose
sudo easy_install ghp-import
sudo apt-get install pyflakes
sudo apt-get install pychecker
sudo easy_install https://github.com/ponty/paved/zipball/master
sudo apt-get install scrot
sudo apt-get install xvfb
sudo apt-get install xserver-xephyr
sudo apt-get install python-imaging
sudo apt-get install python-sphinx
sudo easy_install sphinxcontrib-programsscreenshot
sudo easy_install sphinxcontrib-programoutput
sudo easy_install sphinxcontrib-paverutils
```

7.3 Tasks

`Paver` is used for task management, settings are saved in `pavement.py`. `Sphinx` is used to generate documentation.

print [paver](#) settings:

```
paver printoptions
```

clean generated files:

```
paver clean
```

generate documentation under *docs/_build/html*:

```
paver cog pdf html
```

upload documentation to [github](#):

```
paver ghpages
```

run unit tests:

```
paver nose  
#or  
nosetests --verbose
```

check python code:

```
paver pyflakes  
paver pychecker
```

generate python distribution:

```
paver sdist
```

upload python distribution to [PyPI](#):

```
paver upload
```

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

p

- `pysimavr.ac`, 15
- `pysimavr.avr`, 15
- `pysimavr.connect`, 15
- `pysimavr.firmware`, 15
- `pysimavr.inverter`, 15
- `pysimavr.lcd`, 15
- `pysimavr.ledrow`, 16
- `pysimavr.sgm7`, 16
- `pysimavr.swig.ac_input`, 9
- `pysimavr.swig.hd44780`, 9
- `pysimavr.swig.inverter`, 9
- `pysimavr.swig.ledrow`, 9
- `pysimavr.swig.sgm7`, 10
- `pysimavr.swig.simavr`, 10
- `pysimavr.vcdfile`, 16

INDEX

A

Ac (class in pysimavr.ac), 15
ac_input_t (class in pysimavr.swig.ac_input), 9
add_signal() (pysimavr.vcdfile.VcdFile method), 16
addr (pysimavr.swig.simavr.avr_symbol_t attribute), 11
addr (pysimavr.swig.simavr.elf_firmware_t_trace attribute), 14
alias (pysimavr.swig.simavr.avr_vcd_signal_t attribute), 13
arduino_targets (pysimavr.avr.Avr attribute), 15
aref (pysimavr.swig.simavr.avr_t attribute), 11
aref (pysimavr.swig.simavr.elf_firmware_t attribute), 14
avcc (pysimavr.avr.Avr attribute), 15
avcc (pysimavr.swig.simavr.avr_t attribute), 11
avcc (pysimavr.swig.simavr.elf_firmware_t attribute), 14
Avr (class in pysimavr.avr), 15
avr (pysimavr.swig.ac_input.ac_input_t attribute), 9
avr (pysimavr.swig.hd44780.hd44780_t attribute), 9
avr (pysimavr.swig.inverter.inverter_t attribute), 9
avr (pysimavr.swig.ledrow.ledrow_t attribute), 9
avr (pysimavr.swig.sgm7.sgm7_t attribute), 10
avr (pysimavr.swig.simavr.avr_io_t attribute), 10
avr (pysimavr.swig.simavr.avr_vcd_t attribute), 14
avr_io_t (class in pysimavr.swig.simavr), 10
avr_iopin_t (class in pysimavr.swig.simavr), 10
avr_ioport_getirq_t (class in pysimavr.swig.simavr), 10
avr_ioport_state_t (class in pysimavr.swig.simavr), 10
avr_ioport_t (class in pysimavr.swig.simavr), 11
avr_irq_pool_t (class in pysimavr.swig.simavr), 11
avr_irq_t (class in pysimavr.swig.simavr), 11
avr_kind_t (class in pysimavr.swig.simavr), 11
avr_symbol_t (class in pysimavr.swig.simavr), 11
avr_t (class in pysimavr.swig.simavr), 11
avr_t_cycle_timer (class in pysimavr.swig.simavr), 12
avr_t_io (class in pysimavr.swig.simavr), 13
avr_t_io_r (class in pysimavr.swig.simavr), 13
avr_t_io_shared_io (class in pysimavr.swig.simavr), 13
avr_t_io_shared_io_io (class in pysimavr.swig.simavr), 13
avr_t_io_w (class in pysimavr.swig.simavr), 13
avr_trace_data_t (class in pysimavr.swig.simavr), 13
avr_trace_data_t_old (class in pysimavr.swig.simavr), 13

avr_vcd_log_t (class in pysimavr.swig.simavr), 13
avr_vcd_signal_t (class in pysimavr.swig.simavr), 13
avr_vcd_t (class in pysimavr.swig.simavr), 14

B

bit (pysimavr.swig.simavr.avr_ioport_getirq_t attribute), 10
bsssize (pysimavr.swig.simavr.elf_firmware_t attribute), 14

C

c (pysimavr.swig.simavr.avr_t_io_r attribute), 13
c (pysimavr.swig.simavr.avr_t_io_shared_io_io attribute), 13
c (pysimavr.swig.simavr.avr_t_io_w attribute), 13
codeend (pysimavr.swig.simavr.avr_t attribute), 11
codeline (pysimavr.swig.simavr.avr_trace_data_t attribute), 13
codeline (pysimavr.swig.simavr.elf_firmware_t attribute), 14
codesize (pysimavr.swig.simavr.elf_firmware_t attribute), 14
command_register_addr (pysimavr.swig.simavr.elf_firmware_t attribute), 14
connect_irqs() (in module pysimavr.connect), 15
connect_pins_by_rule() (in module pysimavr.connect), 15
console_register_addr (pysimavr.swig.simavr.elf_firmware_t attribute), 14
count (pysimavr.swig.simavr.avr_irq_pool_t attribute), 11
cursor (pysimavr.swig.hd44780.hd44780_t attribute), 9
cycle (pysimavr.swig.simavr.avr_t attribute), 11
cycle_timer (pysimavr.swig.simavr.avr_t attribute), 11
cycle_timer_map (pysimavr.swig.simavr.avr_t attribute), 11

D

data (pysimavr.swig.simavr.avr_t attribute), 11
datapins (pysimavr.swig.hd44780.hd44780_t attribute), 9
datasize (pysimavr.swig.simavr.elf_firmware_t attribute), 14

- ddr (pysimavr.swig.simavr.avr_ioport_state_t attribute), 10
 - dealloc (pysimavr.swig.simavr.avr_io_t attribute), 10
 - digit_count (pysimavr.swig.sgm7.sgm7_t attribute), 10
 - digit_pin (pysimavr.swig.sgm7.sgm7_t attribute), 10
 - digit_port (pysimavr.swig.sgm7.sgm7_t attribute), 10
 - digit_segments (pysimavr.swig.sgm7.sgm7_t attribute), 10
 - digit_segments() (pysimavr.sgm7.Sgm7 method), 16
 - digit_segments_changed (pysimavr.swig.sgm7.sgm7_t attribute), 10
- ## E
- e2end (pysimavr.swig.simavr.avr_t attribute), 11
 - eeeprom (pysimavr.swig.simavr.elf_firmware_t attribute), 14
 - eesize (pysimavr.swig.simavr.elf_firmware_t attribute), 14
 - eind (pysimavr.swig.simavr.avr_t attribute), 11
 - elf_firmware_t (class in pysimavr.swig.simavr), 14
 - elf_firmware_t_trace (class in pysimavr.swig.simavr), 14
- ## F
- filename (pysimavr.swig.simavr.avr_vcd_t attribute), 14
 - Firmware (class in pysimavr.firmware), 15
 - flags (pysimavr.swig.hd44780.hd44780_t attribute), 9
 - flags (pysimavr.swig.simavr.avr_irq_t attribute), 11
 - flash (pysimavr.swig.simavr.avr_t attribute), 11
 - flash (pysimavr.swig.simavr.elf_firmware_t attribute), 14
 - flashbase (pysimavr.swig.simavr.elf_firmware_t attribute), 14
 - flashend (pysimavr.swig.simavr.avr_t attribute), 12
 - flashsize (pysimavr.swig.simavr.elf_firmware_t attribute), 14
 - fpeek() (pysimavr.avr.Avr method), 15
 - frequency (pysimavr.swig.simavr.avr_t attribute), 12
 - frequency (pysimavr.swig.simavr.elf_firmware_t attribute), 14
 - fuse (pysimavr.swig.simavr.avr_t attribute), 12
- ## G
- gdb (pysimavr.swig.simavr.avr_t attribute), 12
 - gdb_port (pysimavr.swig.simavr.avr_t attribute), 12
 - get_char() (pysimavr.lcd.Lcd method), 15
 - getirq() (pysimavr.ac.Ac method), 15
 - getirq() (pysimavr.avr.Avr method), 15
 - getirq() (pysimavr.inverter.Inverter method), 15
 - getirq() (pysimavr.lcd.Lcd method), 15
 - getirq() (pysimavr.ledrow.LedRow method), 16
 - getirq() (pysimavr.sgm7.Sgm7 method), 16
 - goto_cycle() (pysimavr.avr.Avr method), 15
 - goto_time() (pysimavr.avr.Avr method), 15
- ## H
- h (pysimavr.swig.hd44780.hd44780_t attribute), 9
 - hd44780_t (class in pysimavr.swig.hd44780), 9
 - hook (pysimavr.swig.simavr.avr_irq_t attribute), 11
- ## I
- i_shadow (pysimavr.swig.simavr.avr_t attribute), 12
 - init (pysimavr.swig.simavr.avr_t attribute), 12
 - Inverter (class in pysimavr.inverter), 15
 - inverter_t (class in pysimavr.swig.inverter), 9
 - io (pysimavr.swig.simavr.avr_ioport_t attribute), 11
 - io (pysimavr.swig.simavr.avr_t attribute), 12
 - io (pysimavr.swig.simavr.avr_t_io_shared_io attribute), 13
 - io_port (pysimavr.swig.simavr.avr_t attribute), 12
 - io_shared_io (pysimavr.swig.simavr.avr_t attribute), 12
 - io_shared_io_count (pysimavr.swig.simavr.avr_t attribute), 12
 - ioctl (pysimavr.swig.simavr.avr_io_t attribute), 10
 - irq (pysimavr.swig.ac_input.ac_input_t attribute), 9
 - irq (pysimavr.swig.hd44780.hd44780_t attribute), 9
 - irq (pysimavr.swig.inverter.inverter_t attribute), 9
 - irq (pysimavr.swig.ledrow.ledrow_t attribute), 10
 - irq (pysimavr.swig.sgm7.sgm7_t attribute), 10
 - irq (pysimavr.swig.simavr.avr_io_t attribute), 10
 - irq (pysimavr.swig.simavr.avr_ioport_getirq_t attribute), 10
 - irq (pysimavr.swig.simavr.avr_irq_pool_t attribute), 11
 - irq (pysimavr.swig.simavr.avr_irq_t attribute), 11
 - irq (pysimavr.swig.simavr.avr_t_io attribute), 13
 - irq (pysimavr.swig.simavr.avr_vcd_signal_t attribute), 13
 - irq_count (pysimavr.swig.simavr.avr_io_t attribute), 10
 - irq_ioctl_get (pysimavr.swig.simavr.avr_io_t attribute), 10
 - irq_names (pysimavr.swig.simavr.avr_io_t attribute), 10
 - irq_pool (pysimavr.swig.simavr.avr_t attribute), 12
- ## K
- kind (pysimavr.swig.simavr.avr_io_t attribute), 10
- ## L
- Lcd (class in pysimavr.lcd), 15
 - LedRow (class in pysimavr.ledrow), 16
 - ledrow_t (class in pysimavr.swig.ledrow), 9
 - load_firmware() (pysimavr.avr.Avr method), 15
 - log (pysimavr.swig.simavr.avr_t attribute), 12
 - log (pysimavr.swig.simavr.avr_vcd_t attribute), 14
 - logindex (pysimavr.swig.simavr.avr_vcd_t attribute), 14
- ## M
- make (pysimavr.swig.simavr.avr_kind_t attribute), 11
 - mask (pysimavr.swig.simavr.elf_firmware_t_trace attribute), 14
 - mcu (pysimavr.firmware.Firmware attribute), 15
 - mmcu (pysimavr.swig.simavr.avr_t attribute), 12
 - mmcu (pysimavr.swig.simavr.elf_firmware_t attribute), 14

move_time_marker() (pysimavr.avr.Avr method), 15

N

name (pysimavr.swig.simavr.avr_ioport_state_t attribute), 10

name (pysimavr.swig.simavr.avr_ioport_t attribute), 11

name (pysimavr.swig.simavr.avr_irq_t attribute), 11

name (pysimavr.swig.simavr.avr_vcd_signal_t attribute), 13

name (pysimavr.swig.simavr.elf_firmware_t_trace attribute), 14

names (pysimavr.swig.simavr.avr_kind_t attribute), 11

next (pysimavr.swig.simavr.avr_io_t attribute), 10

next_cycle_timer (pysimavr.swig.simavr.avr_t attribute), 12

O

old (pysimavr.swig.simavr.avr_trace_data_t attribute), 13

old_pci (pysimavr.swig.simavr.avr_trace_data_t attribute), 13

out (pysimavr.swig.inverter.inverter_t attribute), 9

out() (pysimavr.inverter.Inverter method), 15

output (pysimavr.swig.simavr.avr_vcd_t attribute), 14

P

param (pysimavr.swig.simavr.avr_t_cycle_timer attribute), 12

param (pysimavr.swig.simavr.avr_t_io_r attribute), 13

param (pysimavr.swig.simavr.avr_t_io_shared_io_io attribute), 13

param (pysimavr.swig.simavr.avr_t_io_w attribute), 13

pause() (pysimavr.avr.Avr method), 15

pc (pysimavr.swig.simavr.avr_t attribute), 12

pc (pysimavr.swig.simavr.avr_trace_data_t_old attribute), 13

pcint (pysimavr.swig.simavr.avr_ioport_t attribute), 11

peek() (pysimavr.avr.Avr method), 15

pending (pysimavr.swig.simavr.avr_t attribute), 12

pending_wait (pysimavr.swig.simavr.avr_t attribute), 12

period (pysimavr.swig.simavr.avr_vcd_t attribute), 14

pin (pysimavr.swig.simavr.avr_iopin_t attribute), 10

pin (pysimavr.swig.simavr.avr_ioport_state_t attribute), 10

pinindex() (pysimavr.sgm7.Sgm7 method), 16

pinstate (pysimavr.swig.hd44780.hd44780_t attribute), 9

pinstate (pysimavr.swig.ledrow.ledrow_t attribute), 10

pinstate (pysimavr.swig.sgm7.sgm7_t attribute), 10

pinstate() (pysimavr.lcd.Lcd method), 16

pinstate() (pysimavr.ledrow.LedRow method), 16

pinstate() (pysimavr.sgm7.Sgm7 method), 16

pinstate_changed (pysimavr.swig.ledrow.ledrow_t attribute), 10

pool (pysimavr.swig.simavr.avr_irq_t attribute), 11

port (pysimavr.swig.simavr.avr_iopin_t attribute), 10

port (pysimavr.swig.simavr.avr_ioport_state_t attribute), 11

pysimavr.ac (module), 15

pysimavr.avr (module), 15

pysimavr.connect (module), 15

pysimavr.firmware (module), 15

pysimavr.inverter (module), 15

pysimavr.lcd (module), 15

pysimavr.ledrow (module), 16

pysimavr.sgm7 (module), 16

pysimavr.swig.ac_input (module), 9

pysimavr.swig.hd44780 (module), 9

pysimavr.swig.inverter (module), 9

pysimavr.swig.ledrow (module), 9

pysimavr.swig.sgm7 (module), 10

pysimavr.swig.simavr (module), 10

pysimavr.vcdfile (module), 16

R

r (pysimavr.swig.simavr.avr_t_io attribute), 13

r_ddr (pysimavr.swig.simavr.avr_ioport_t attribute), 11

r_pcint (pysimavr.swig.simavr.avr_ioport_t attribute), 11

r_pin (pysimavr.swig.simavr.avr_ioport_t attribute), 11

r_port (pysimavr.swig.simavr.avr_ioport_t attribute), 11

ramend (pysimavr.swig.simavr.avr_t attribute), 12

rampz (pysimavr.swig.simavr.avr_t attribute), 12

read() (pysimavr.firmware.Firmware method), 15

readpins (pysimavr.swig.hd44780.hd44780_t attribute), 9

reset (pysimavr.swig.simavr.avr_io_t attribute), 10

reset (pysimavr.swig.simavr.avr_t attribute), 12

reset() (pysimavr.avr.Avr method), 15

reset() (pysimavr.lcd.Lcd method), 16

reset_dirty() (pysimavr.ledrow.LedRow method), 16

reset_dirty() (pysimavr.sgm7.Sgm7 method), 16

run (pysimavr.swig.simavr.avr_t attribute), 12

run() (pysimavr.avr.Avr method), 15

S

segment_pin (pysimavr.swig.sgm7.sgm7_t attribute), 10

segment_port (pysimavr.swig.sgm7.sgm7_t attribute), 10

Sgm7 (class in pysimavr.sgm7), 16

sgm7_t (class in pysimavr.swig.sgm7), 10

signal (pysimavr.swig.simavr.avr_vcd_log_t attribute), 13

signal (pysimavr.swig.simavr.avr_vcd_t attribute), 14

signal_count (pysimavr.swig.simavr.avr_vcd_t attribute), 14

signature (pysimavr.swig.simavr.avr_t attribute), 12

size (pysimavr.swig.simavr.avr_vcd_signal_t attribute), 13

sleep (pysimavr.swig.simavr.avr_t attribute), 12

sp (pysimavr.swig.simavr.avr_trace_data_t_old attribute), 13

special_deinit (pysimavr.swig.simavr.avr_t attribute),
12
special_init (pysimavr.swig.simavr.avr_t attribute), 12
sreg (pysimavr.swig.simavr.avr_t attribute), 12
start (pysimavr.swig.simavr.avr_vcd_t attribute), 14
start() (pysimavr.vcdfile.VcdFile method), 16
state (pysimavr.swig.simavr.avr_t attribute), 12
states (pysimavr.avr.Avr attribute), 15
step() (pysimavr.avr.Avr method), 15
stop() (pysimavr.vcdfile.VcdFile method), 16
symbol (pysimavr.swig.simavr.avr_symbol_t attribute),
11

T

terminate() (pysimavr.avr.Avr method), 15
terminate() (pysimavr.vcdfile.VcdFile method), 16
time_passed() (pysimavr.avr.Avr method), 15
timer (pysimavr.swig.simavr.avr_t_cycle_timer attribute), 12
touched (pysimavr.swig.simavr.avr_trace_data_t attribute), 13
trace (pysimavr.swig.simavr.avr_t attribute), 12
trace (pysimavr.swig.simavr.elf_firmware_t attribute),
14
trace_data (pysimavr.swig.simavr.avr_t attribute), 12
tracecount (pysimavr.swig.simavr.elf_firmware_t attribute), 14
tracename (pysimavr.swig.simavr.elf_firmware_t attribute), 14
traceperiod (pysimavr.swig.simavr.elf_firmware_t attribute), 14

U

UnkownAvrError, 15
used (pysimavr.swig.simavr.avr_t_io_shared_io attribute), 13

V

value (pysimavr.swig.ac_input.ac_input_t attribute), 9
value (pysimavr.swig.simavr.avr_irq_t attribute), 11
value (pysimavr.swig.simavr.avr_vcd_log_t attribute),
13
vcc (pysimavr.avr.Avr attribute), 15
vcc (pysimavr.swig.simavr.avr_t attribute), 12
vcc (pysimavr.swig.simavr.elf_firmware_t attribute), 14
vcd (pysimavr.swig.simavr.avr_t attribute), 12
VcdFile (class in pysimavr.vcdfile), 16
vector (pysimavr.swig.simavr.avr_t attribute), 12
vector_size (pysimavr.swig.simavr.avr_t attribute), 12
vram (pysimavr.swig.hd44780.hd44780_t attribute), 9

W

w (pysimavr.swig.hd44780.hd44780_t attribute), 9
w (pysimavr.swig.simavr.avr_t_io attribute), 13
when (pysimavr.swig.simavr.avr_t_cycle_timer attribute), 12
when (pysimavr.swig.simavr.avr_vcd_log_t attribute),
13